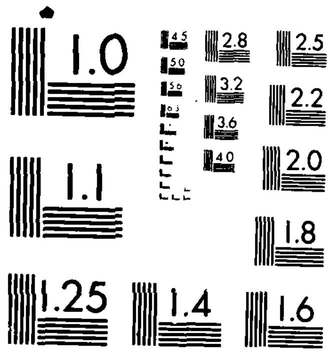
END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

REPORT ACT-61 JUNE 1985

12

ORDINATED

APPLIED COMPUTATION

AD-A161 345

A NETWORK FLOW APPROACH TO THE WAFER SCALE INTEGRATION OF VLSI ARRAYS

UILU-ENG 85-2222

UNIVERSITY OF ILLINOIS

11 18 85 035

AD-A161 345

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS: None
2a. SECURITY CLASSIFICATION AUTHORITY: N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release, Distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): R-1047 UILU-Eng 85 - 2222 (ACT-61)	5. MONITORING ORGANIZATION REPORT NUMBER(S): N/A
6a. NAME OF PERFORMING ORGANIZATION: Coordinated Science Lab. Univ. of Illinois; 6b. OFFICE SYMBOL (If applicable): N/A	7a. NAME OF MONITORING ORGANIZATION: Joint Services Electronics Program
6c. ADDRESS (City, State and ZIP Code): 1101 W. Springfield Avenue Urbana, Illinois 61801	7b. ADDRESS (City, State and ZIP Code): 800 N. Quincy Arlington, VA 22217
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: JSEP; 8b. OFFICE SYMBOL (If applicable): N/A	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: N00014 - 84 - C -0149
8c. ADDRESS (City, State and ZIP Code): 800 N. Quincy Arlington, VA 22217	10. SOURCE OF FUNDING NOS.: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification) A Network Flow Approach to the Wafer Scale Integration of VLSI Arrays	
12. PERSONAL AUTHOR(S): Bruno Codenotti & Roberto Tamassia	
13a. TYPE OF REPORT: Technical; 13b. TIME COVERED FROM -- TO --	14. DATE OF REPORT (Yr., Mo., Day): June 1985; 15. PAGE COUNT: 15
16. SUPPLEMENTARY NOTATION	
17. COSATI CODES: FIELD / GROUP / SUB. GR.	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): Network Flow, VLSI Array, Fault Tolerance

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

An algorithm is described for reconfiguring a 2-dimensional VLSI array on a silicon wafer that has some faulty cells. The functional cells of the array are interconnected in order to simulate a fault-free array of smaller size, where the interconnection wires are routed inside horizontal and vertical channels, according to the Manhattan model. The concept of simulation distance is introduced, and it is shown to be related to the length of the longest interconnection wire. The algorithm makes use of network flow techniques in order to find a wiring with minimum simulation distance. This results in a practical heuristic for minimizing the maximum wire length. The complexity and performance of this algorithm are also discussed in the paper

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL: None

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

SECURITY CLASSIFICATION OF THIS PAGE

A NETWORK FLOW APPROACH TO THE WAFER SCALE INTEGRATION OF VLSI ARRAYS [*]

by

Bruno Codenotti [†] and *Roberto Tamassia* [‡]

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

[*] This research was partially supported by the Joint Services Electronics Program under Contract N00014-84-C-0149. Roberto Tamassia was partially supported by a Fulbright grant.

[†] On leave from: Istituto di Elaborazione dell'Informazione CNR Via Santa Maria 46 56100 Pisa, Italy.

[‡] On leave from: Dipartimento di Informatica e Sistemistica Universita' di Roma "La Sapienza" Via Buonarroti 12 00185 Roma, Italy.

ABSTRACT

An algorithm is described for reconfiguring a 2-dimensional VLSI array on a silicon wafer that has some faulty cells. The functional cells of the array are interconnected in order to simulate a fault-free array of smaller size, where the interconnection wires are routed inside horizontal and vertical channels, according to the Manhattan model. The concept of *simulation distance* is introduced, and it is shown to be related to the length of the longest interconnection wire. The algorithm makes use of network flow techniques in order to find a wiring with minimum simulation distance. This results in a practical heuristic for minimizing the maximum wire length. The complexity and performance of this algorithm are also discussed in the paper.

KEYWORDS: network flow, VLSI array, fault tolerance.

1. INTRODUCTION

The technique of *wafer scale integration* for VLSI circuits has received considerable attention in recent years. The basic idea is to assemble a large system of processors, or *cells*, on a single silicon wafer so that the chip packaging costs are cut off. Due to the physical and technological limits of the integration process, some cells of the wafer can be defective, or "dead". Therefore, the problem arises of reconfiguring the interconnection network using the "live" cells.

The reconfiguration of one and two-dimensional arrays of cells, typical of VLSI systolic architectures, has been investigated in [4, 2]. In both papers, a probabilistic model of cell failure is adopted and algorithms for minimizing the maximum wire length in the reconstructed array are given. Channel width and area penalties are also considered.

In this paper we present a new approach to the problem of reconfiguring two-dimensional arrays, based on network flow techniques. Section 2 illustrates the basic ideas on which our approach is based. In Section 3, we give an algorithm for the reconstruction of two-dimensional arrays. Section 4 contains some conclusive remarks.

2. PRELIMINARIES

We assume the same model for the wafer scale system as in [4]. The cells are positioned in a $\sqrt{n} \times \sqrt{n}$ array. The strips between two rows or columns of cells are called *channels*. Each channel contains a fixed number of tracks on which the interconnection wires are routed. Suppose that m cells in the array are dead. We investigate the problem of interconnecting the remaining live cells into a square array of size $\lfloor\sqrt{n-m}\rfloor \times \lfloor\sqrt{n-m}\rfloor$, with the goal of minimizing the maximum wire length in the reconstructed array. See an example in figs. 1 and 2, taken from [4].

Consider the distribution of faults shown in fig. 3. It is easy to see that the wiring of fig. 4 provides an optimal reconstruction. This observation suggests the following reconfiguration strategy:

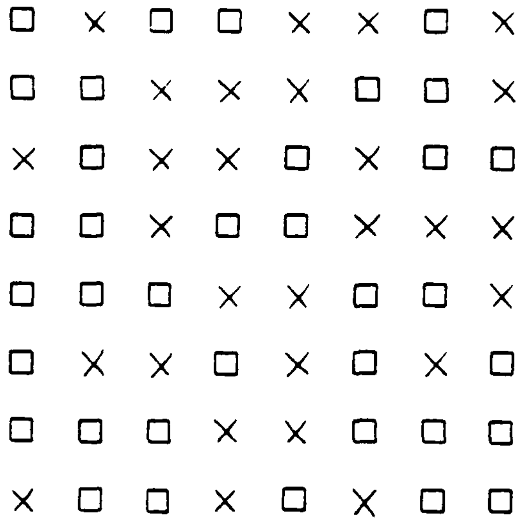


Figure 1 A square array of 64 cells. The live cells are represented by a square. The dead cells are represented by a cross.

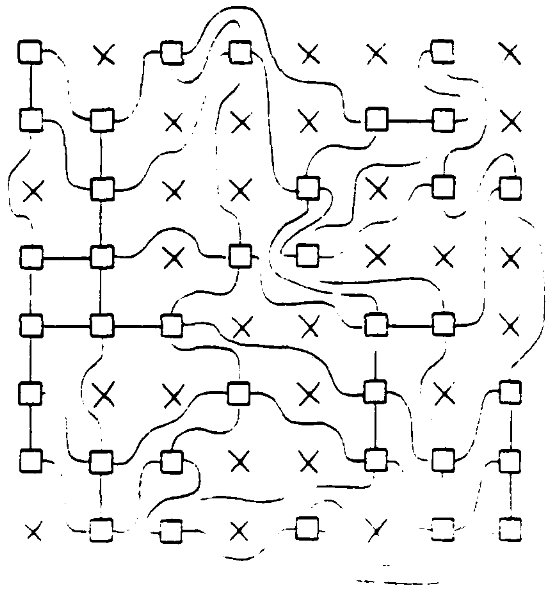


Figure 2 Example of reconfiguration of the array in fig. 1, so that the live cells form a square array of 36 cells.

(1) Select an *intermediate array* whose dead cell distribution consists of exactly $\sqrt{n} - \sqrt{n-m}$ rows and $\sqrt{n} - \sqrt{n-m}$ columns.

(2) Interconnect the live cells of the given array in order to construct the intermediate array

(3) Route the intermediate array by connecting each live cell to the closest live cells on the same row or column.

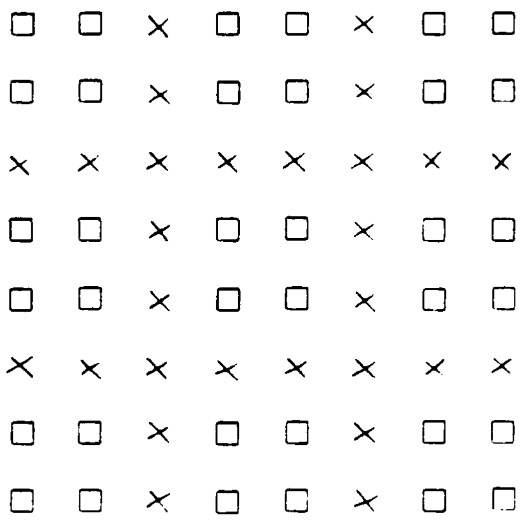


Figure 3 An array of 64 cells whose dead cells are arranged into two rows and two columns.

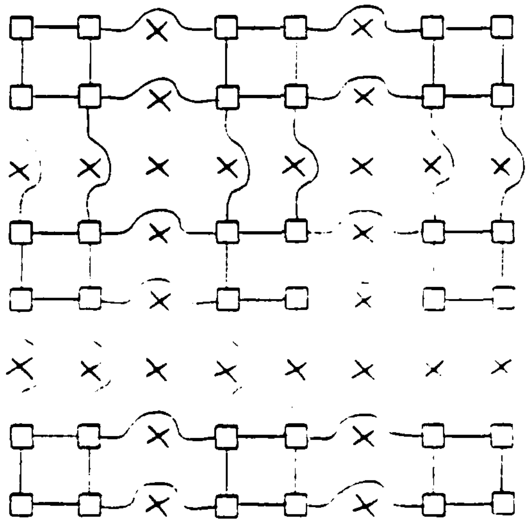


Figure 4 Optimal reconfiguration of the array in fig 3.

The second step can be viewed as a *simulation* of the intermediate array by the original one. Namely, each live cell of the intermediate array is simulated by a distinct live cell of the original array.

We introduce now the basic terminology on network flows, to be used in the next sections.

A (*flow*) *network* is a 6-tuple $N = (U, A, b, c, s, t)$ where:

(1) U is a set of *nodes*;

(2) $A \subseteq U \times U$ is a set of directed *arcs* (the digraph with vertex set V and arc set A is called *underlying digraph* of N);

(3) $b : U \bigcup A \to \mathbf{N}$ associates to all nodes and arcs a nonnegative integer *capacity*;

(4) $c : A \to \mathbf{N}$ associates to each arc a nonnegative integer *cost*;

(5) s and t are two designated nodes called the *source* and the *sink*, respectively.

A *flow* for N is a function $f : A \to \mathbf{R}$ that satisfies the following conditions:

(1) $0 \leqslant f(u,v) \leqslant b(u,v)$ for all $(u,v) \in A$;

(2) $\sum_{u : (u,v) \in A} f(u,v) = \sum_{w : (v,w) \in A} f(v,w) \leqslant b(v)$ for all $v \in U - \{s,t\}$.

The cost of the flow f is the quantity:

$$\mathrm{COST}(f) = \sum_{(u,v) \in A} c(u,v) \cdot f(u,v).$$

The value of the flow f is given by:

$$\mathrm{VALUE}(f) = \sum_{v : (s,v) \in A} f(s,v).$$

A *minimum cost flow of value* ϕ for N is a flow f for N such that $\mathrm{VALUE}(f) = \phi$ and $\mathrm{COST}(f)$ is minimum.

3. THE NETWORK FLOW TECHNIQUE

We represent the 2-dimensional array with an undirected graph $G = (V, E)$, where:

$$V = \{(i,j) \mid 1 \leqslant i,j \leqslant \sqrt{n}\}, \text{ and}$$

$$E = \{((i,j),(k,l)) \mid |i-k| + |j-l| = 1\}.$$

Let $d_G(u,v)$ denote the distance between vertices u and v in the graph G. The dead cells are in a subset of V, denoted by D.

In the rest of this section, we assume that the intermediate array is given and we denote by T the set of its dead cells, which we call *target*. Note that $|T| \geqslant |D|$. The simulation between the cells of the original and intermediate array is expressed by a function σ mapping vertices into vertices where $\sigma(u) = v$ iff cell v of the intermediate array is simulated by cell u of the original array. The function σ must satisfy the following properties:

(1) no cell is simulated by more than one cell;

(2) only the live cells can perform simulations;

(3) no dead cell in the target is simulated;

(4) all cells outside the target have to be simulated.

More formally, this can be expressed by defining σ as a one-to-one function (property 1) mapping the set $V - D$ (property 2) into the set $V - (D \bigcap T)$ (property 3), such that, for all $v \in V - T$, there exists a vertex u such that $\sigma(u) = v$ (property 4). We associate to σ a directed graph $G_\sigma = (V_\sigma, A_\sigma)$, where $V_\sigma = V - (D \bigcap T)$ and $(u,v) \in A_\sigma$ whenever $\sigma(u) = v$.

Lemma 1 A function σ is a simulation mapping if and only if the graph G_σ consists of disjoint paths that are either cycles of one or more live cells or chains starting from a live cell in the target, ending at a dead cell outside the target, and having the intermediate cells all live. Furthermore, the number of these chains is exactly $|D \bigcap (V - T)|$.

Proof: Assume that σ is a simulation mapping. Since σ is injective, each vertex in V_σ has at most one ingoing and one outgoing arc. This implies that G_σ consists of disjoint cycles and/or chains. Because σ is defined on the set of live cells, then each cycle in G_σ contains only live cells and every chain in G_σ consists of all live cells, but the last one, which is a dead cell outside the target. Furthermore, all chains start at a live cell inside the target, since each vertex in $V - T$ has an inverse mapping. Finally, each dead cell outside the target has an ingoing arc, so that the number of chains is exactly $|D \bigcap (V - T)|$. This completes the proof of the *Only-If* part. The proof of the *If* part readily follows. □

Let $d_G(u,v)$ denote the distance between vertices u and v in the graph G, i.e. the minimum number of edges in any path from u to v. The *maximum wire length* l_σ in the interconnection of the intermediate array generated by σ is given by:

$$l_\sigma = \max \{d_G(u,v) \mid (\sigma(u), \sigma(v)) \in E\}.$$

We introduce now the concept of *simulation distance* d_σ, defined by:

$$d_\sigma = \max \{d_G(u, \sigma(u)) \mid u \in V_\sigma\}.$$

which is related to the maximum wire length as stated in the following proposition.

Proposition 1 Let σ be a simulation mapping, then $l_\sigma \leqslant 2d_\sigma + 1$.

Proof: A consequence of the inequality: $d_G(u,v) \leqslant d_G(u,w) + d_G(w,v)$. □

Given a positive integer k, an auxiliary graph can be used to check whether $d_\sigma \leqslant k$. Let $G^{(k)} = (U, A^{(k)})$ be the directed graph defined as follows.

(1) $U = V \bigcup \{s, t\}$, where s and t are two new distinct vertices.

(2) $A^{(k)} = A_s \bigcup A_m^{(k)} \bigcup A_t$, where:

$$A_s = \{(s, u) \mid u \in T \bigcap (V - D)\},$$

$$A_m^{(k)} = \{(u, v) \in V \times V \mid u \in V - D, v \in V - (D \bigcap T), \text{ and } d_G(u, v) \leqslant k\}, \text{ and}$$

$$A_t = \{(u,t) \mid u \in D \bigcap (V - T)\}.$$

The *length* of arc (u,v) in $A_m^{(k)}$ is defined as the distance between u and v in G. Fig. 5 shows the digraph $G^{(1)}$ corresponding to the array of fig. 1 and to the target of fig. 3.

Proposition 2 Let G, D and T be an instance of the simulation problem. Then there exists a simulation function σ with simulation distance $d_\sigma = k$ if and only if the digraph $G^{(k)}$ contains $|D \bigcap (V - T)|$ disjoint directed paths from s to t.

Proof: Follows from the definition of $G^{(k)}$ and Lemma 1. □

In order to minimize the global wire length in the reconstruction of the intermediate array, we look for a set of disjoint paths in $G^{(k)}$ with minimum global length. Moreover, it is convenient to choose the arcs in each path as short as possible, so to decrease the probability that l_σ attains the bound $2d_\sigma + 1$. For this purpose, we can associate a cost $c(u,v) = \chi(\mathrm{d}_G(u,v))$ to each arc (u,v) in

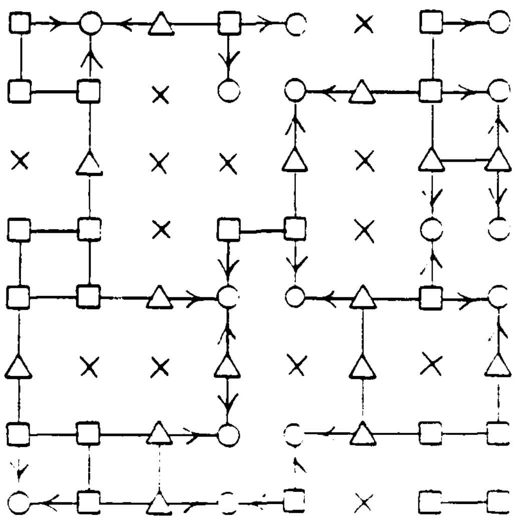


Figure 5 The digraph G^{1} corresponding to the array of fig 1 and to the target of fig 3. The dead cells outside the target are represented by a circle. The live cells inside the target are represented by a triangle. Only the arcs in A_m^{1} are shown, where an arc without arrow stands for a pair of symmetric arcs.

$A^{(k)}$, depending on the length of the arc, so that the following conditions are satisfied:

(1) $\chi(d_1 + \cdots + d_p) > \chi(d_1) + \cdots + \chi(d_p)$, for all $d_i \in \mathbf{N}$, $i = 1, \ldots, p$.

(2) $\chi(c_1) + \cdots + \chi(c_q) < \chi(d_1) + \cdots + \chi(d_p)$ implies
$c_1 + \cdots + c_q < d_1 + \cdots + d_p$, for all $c_i, d_j \in \mathbf{N}$, $i = 1, \ldots, q$, $j = 1, \ldots, p$.

Both conditions are fulfilled by the cost function $\chi(d) = d^{1+\epsilon}$, provided ϵ is a sufficiently small positive constant.

Now, we present the algorithm *SIM* that finds a simulation function σ with minimum simulation distance, given a positive integer n, and the sets D and T of dead and target cells, respectively.

Algorithm *SIM*

Input: positive integer n, sets D and T of dead and target cells, respectively;

Output: simulation function σ with minimum simulation distance.

```
begin
    $\phi := |D \bigcap (V - T)|$;
    $BUILDNET\ (1, N^{(1)})$;
    if $MAXFLOW\ (N^{(1)}) \geqslant \phi$
        then begin
            $f := MINCOSTFLOW\ (N^{(1)}, \phi)$;
            $SIGMA\ (N^{1}, f, \sigma)$
        end
        else begin
            $a := 1$;
            $b := 2\sqrt{n} - 2$;
            $k := \lfloor (a + b)/2 \rfloor$;
```

```
                repeat
                    BUILDNET $(k, N^{(k)})$;
                    if MAXFLOW $(N^{(k)}) \geqslant \phi$
                        then $b := k$
                        else $a := k$;
                    $k := \lfloor (a+b)/2 \rfloor$
                until $a = b$;
                $f :=$ MINCOSTFLOW $(N^{(k)}, \phi)$;
                SIGMA $(N^{(k)}, f, \sigma)$
            end
end.

procedure BUILDNET $(k, N^{(k)})$;
begin
    Build the auxiliary flow network $N^{(k)} = (U, A^{(k)}, b, c, s, t)$
    as follows:
        the underlying digraph of $N^{(k)}$ is $G^{(k)}$;
        each vertex and arc has unit capacity;
        each arc $(u, v)$ in $A_m^{(k)}$ has cost $c(u, v) = (d_G(u, v))^{1+\epsilon}$;
        for all the remaining arcs $c(u, v) = 0$.
end;

procedure SIGMA $(N^{(k)}, f, \sigma)$;
begin
    for each arc $(u, v) \in A_m^{(k)}$ do
        if $f(u, v) = 1$
            then $\sigma(u) := v$;
```

```
end;

function MAXFLOW (N );
begin
    Return the maximum flow value for the network N
end;

function MINCOSTFLOW (N ,$\phi$);
begin
    Return a minimum cost flow of value $\phi$ for the network N.
end;
```

Figs. 6 and 7 show the application of algorithm *SIM* to the array of fig. 1.

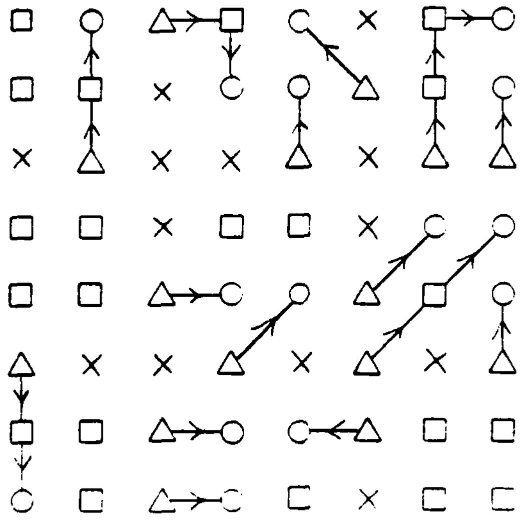


Figure 6 Minimum cost flow in the network $N^{(2)}$ corresponding to the array of fig. 1 and to the target of fig. 3. Only the arcs in $A_m^{(2)}$ with nonzero (unit) flow are shown. Note that $k = 2$ is the minimum value for which $N^{(k)}$ admits a feasible flow of value $|D \bigcap (V - T)| = 15$.

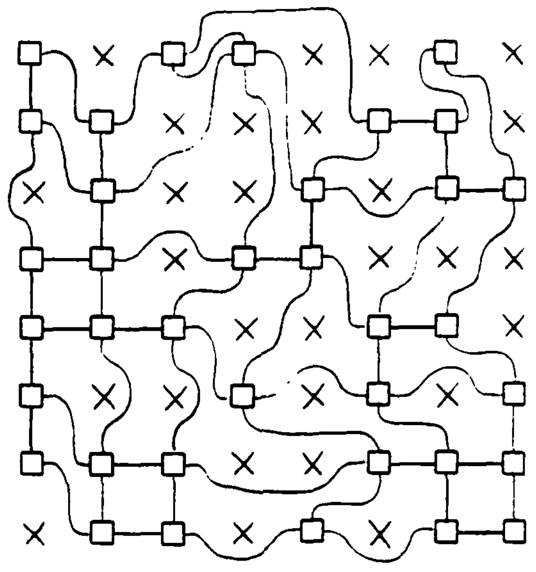


Figure 7 Reconfiguration of the array in fig. 1 produced by the algorithm *SIM*.

Proposition 3 Algorithm *SIM* correctly computes a simulation function σ with minimum simulation distance in $O(n^3)$ time.

Proof: The correctness of the algorithm follows from the fact that any minimum cost flow for N^* of value $\phi = |D \cap (V - T)|$ consists of ϕ disjoint paths from s to t [5].

At most $O(\log n)$ maximum flow computations are performed, each taking time $O(n^{2.5})$ [1, cap. 6]. The minimum cost flow computation is executed only once, and has complexity $O(n^3)$ [3, cap. 4]. □

4. CONCLUDING REMARKS

The algorithm described in the previous section assumes that the target is given. Two criteria seem to be suitable to select a "good" target:

(1) maximum covering of the dead cells;

(2) uniform distribution of the live cells outside the target.

Criterium 1 is preferable if the number of uncovered cells is $o(|D|)$.

A variation of the problem arises when the number of live cells is not a perfect square and all the live cells have to be used in the reconfiguration process. In this case the reconstructed array is a square array that misses some border cells, and the corresponding intermediate array can be constructed with a slight modification of the above technique.

The network approach can be extended in order to take into account also the channel width. Let $N^{(k,c_1,c_2)}$ be the network obtained from $N^{(k)}$ by assigning capacity c_1 to each node and capacity c_2 to each arc in $A_m^{(k)}$. By using arguments similar to the ones of section 2, we obtain that if $N^{(1,c_1,c_2)}$ admits a flow of value $|D \bigcap (V-T)|$, then there is a reconstruction of the array with maximum wire length and channel width bounded by c_1 and c_2, respectively.

ACKNOWLEDGMENT

We wish to thank Franco Preparata for his encouragement and support.

REFERENCES

[1] S. Even, *Graph Algorithms*, Computer Science Press (1979).

[2] J.W. Greene and A. El Gamal, "Configuration of VLSI Arrays in the Presence of Defects," *J. ACM* **31**(4) pp. 694-717 (1984).

[3] E. Lawler, *Combinatorial Optimization*, Holt, Rinehart and Winston (1976).

[4] F.T. Leighton and C.E. Leiserson, "Wafer-Scale Integration of Systolic Arrays," *Proc. 23rd IEEE FOCS*, pp. 297-311 (1982).

[5] K. Menger, "Zur Allgemeinen Kurventheorie," *Fund. Math.* **10** pp. 96-115 (1927).